EDGEWOOD
RESEARCH,
DEVELOPMENT &
ENGINEERING
CENTER

ERDEC-TR-123

THE TRANSPUTER

L. Russell Williams

RESEARCH AND TECHNOLOGY DIRECTORATE

October 1993

U.S. ARMY
CHEMICAL
AND BIOLOGICAL
DEFENSE AGENCY

Aberdeen Proving Ground, Maryland 21010-5423

Disclaimer

The findings in this report are not to be construed as an official Department of the Army position unless so designated by other authorizing documents.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY (Leave blank)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	1993 October	Final, 93 Feb - 93 Jun

4. TITLE AND SUBTITLE

The Transputer

5. FUNDING NUMBERS

PR-1O162622A553

6. AUTHOR(S)

Williams, L. Russell

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

DIR, ERDEC, ATTN: SCBRD-RTM, APG, MD 21010-5423

8. PERFORMING ORGANIZATION REPORT NUMBER

ERDEC-TR-123

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)

10. SPONSORING / MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION / AVAILABILITY STATEMENT

Approved for public release; distribution is unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

In 1984, Inmos introduced the transputer. The transputer is a microprocessor that is designed to operate in parallel with other transputers in the same system. Most aspects of the transputer were designed with this fact in mind. As a result, inter-processor and inter-process communication is a very important part of the design of the transputer. These parallel processing communications abilities can be very valuable to the Army in the area of chemical detection and its subsequent signal processing needs.

14. SUBJECT TERMS

Transputer Parallel computing

15. NUMBER OF PAGES

19

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	UL

Blank

PREFACE

The work described in this report was authorized under Project No. 1O162622A553, CB Defense and General Investigation. This work was started in February 1993 and completed in June 1993.

The use of trade names or manufacturers' names in this report does not constitute an official endorsement of any commercial products. This report may not be cited for purposes of advertisement.

This report has been approved for release to the public. Registered users should request additional copies from the Defense Technical Information Center; unregistered users should direct such requests to the National Technical Information Service.

Acknowledgement

The author thanks Alice M. Harper for input concerning uses of the transputer at the U.S. Army Edgewood Research, Development and Engineering Center.

Blank

CONTENTS

LIST OF FIGURES

LIST OF TABLES

The Transputer

Introduction

The transputer, manufactured by INMOS, is a single chip VLSI device with processor, memory, and communications links. This represents a slight deviation fom current microprocessor technology. The common features of all present transputers are:

- ★ High speed integer processor with microcodes process scheduler.
- ★ On-chip fast static memory.
- ★ Up to four links for communication with other transputers.
- ★ Internal timers.
- ★ External memory interface.

This report is intended to explain the architecture of the transputer and illustrate possible uses in the area of chemical defense. The transputer is already being used in one chemical detector that is under development and is well suited for use in many other chemical defense applications.

Figure 1 on the following page shows the block diagram of a generic transputer. The INMOS transputer is the first single-chip microprocessor to provide a high speed processor, fast inter-processor communications, and explicit support for multiple processes and multiple processor systems. Transputers are designed to be part of a multiprocessor system, so the performance of an individual processor is not especially critical. If more processing power is needed, more processors can simply be added.

The Transputer Family

As can be seen in Table 1, there is a fairly wide range of transputer devices available, but all follow the same basic outline. There are three main groups in the transputer family: the 16-bit T2 series; the 32-bit integer-only T4 series; and the 32-bit T8 series, which has an on-board floating point coprocessor. The T212 is the original 16 bit transputer and has two kilobytes of RAM. The T222 is a more recent device, has more RAM at four kilobytes, and has an extended instruction set, but is otherwise similar to the T212. The T225 is very similar to the T222, but adds extra instructions for debugging. The M212 is derived from the T212, but has a built-in disk interface. It has only two external links -- the other two are used as part of the disk interface.

The T414 is the original member of the transputer family. It is a 32-bit processor with two kilobytes of RAM and no floating point coprocessor. The T425 is an updated version of the T414 and has block move instructions and extensions to support debugging. The T425

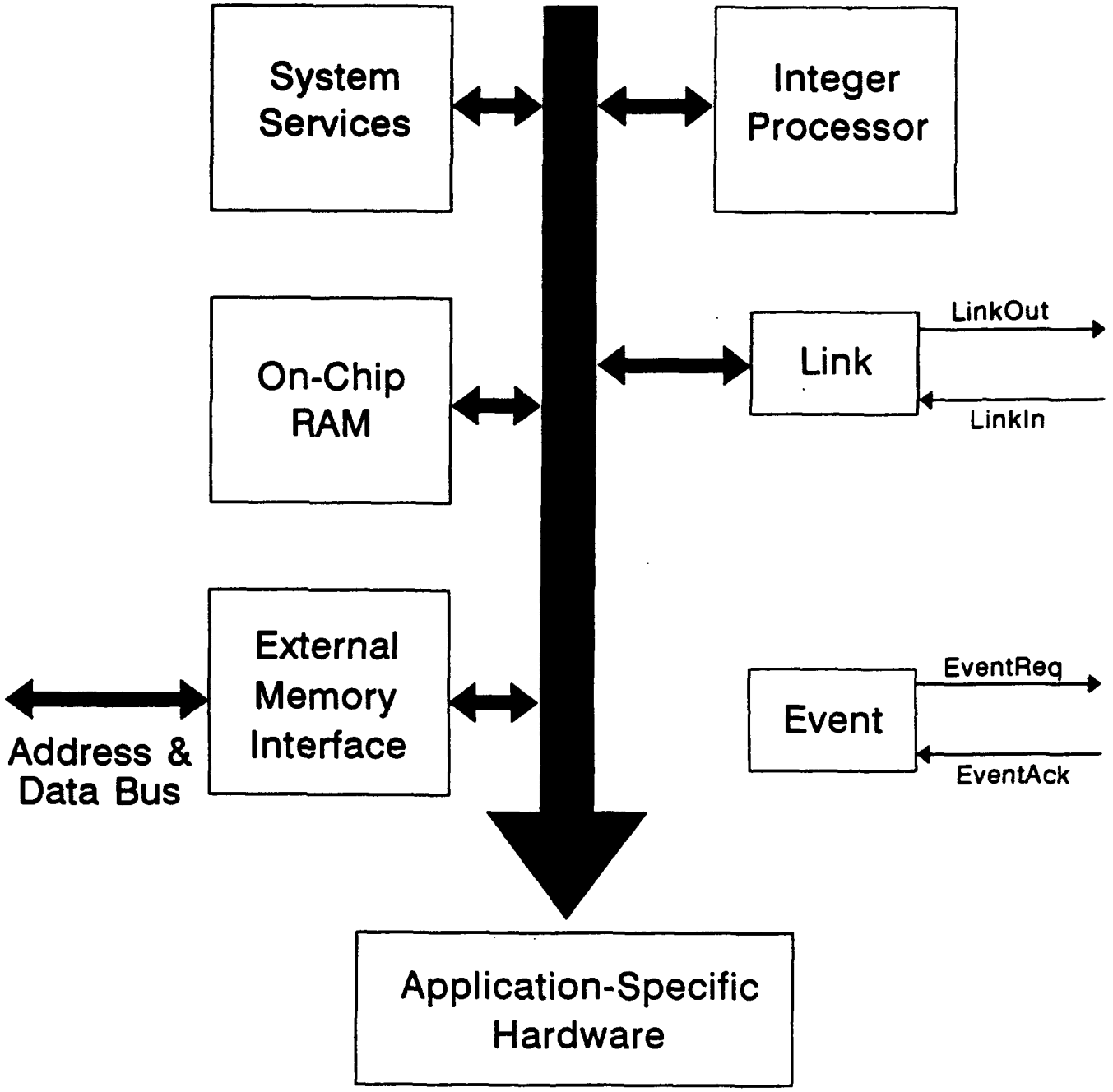

Figure 1: The Generic Transputer

is intended to replace the T414. The T400 is a more recently released, simplified, low-cost version of the T425. The 32-bit T800 was the first transputer specifically designed for numerical applications, so has an on-board floating point coprocessor and four kilobytes of RAM. The T805 is the same as the T800, but has debugging support. The T801 has has a high speed static RAM external memory interface rather than the programmable interface used by other members of the family. This makes it faster, but requires more expensive and less compact memory devices.

The internal design of the transputer is unlike that of any of its predecessors. The central concept of transputer architecture is that of the process. A process represents an individual thread of control and the transputer switches between running processes to provide the illusion that they are all running simultaneously. This is normally handled by the operating system and called multitasking, but in the transputer, this is implemented in hardware and microcoding. All transputers have a fast integer processor and many instructions that take only a single cycle of the processor clock to complete. Transputers currently are manufacured with clock speeds up to 25 MHz. All transputers , however, operate from an external clock speed of 5 MHz. The processor clock is obtained from an internal phase-locked loop multiplier.

16-bit Transputers

	T212	**T222**	**T225**	**M212**
Word Length	16	16	16	16
Internal RAM	2K	4K	4K	2K
Number of Links	4	4	4	2
Extended Instructions	N	Y	Y	N
Debugging Instructions	N	N	Y	N

32-bit Transputers

	T400	**T414**	**T425**	**T800**	**T801**	**T805**
Word Length	32	32	32	32	32	32
Internal RAM	2K	2K	4K	4K	4K	4K
Number of Links	2	4	4	4	4	4
Hardware FPU	N	N	N	Y	Y	Y

Table 1 - The Transputer Family as of December 1989

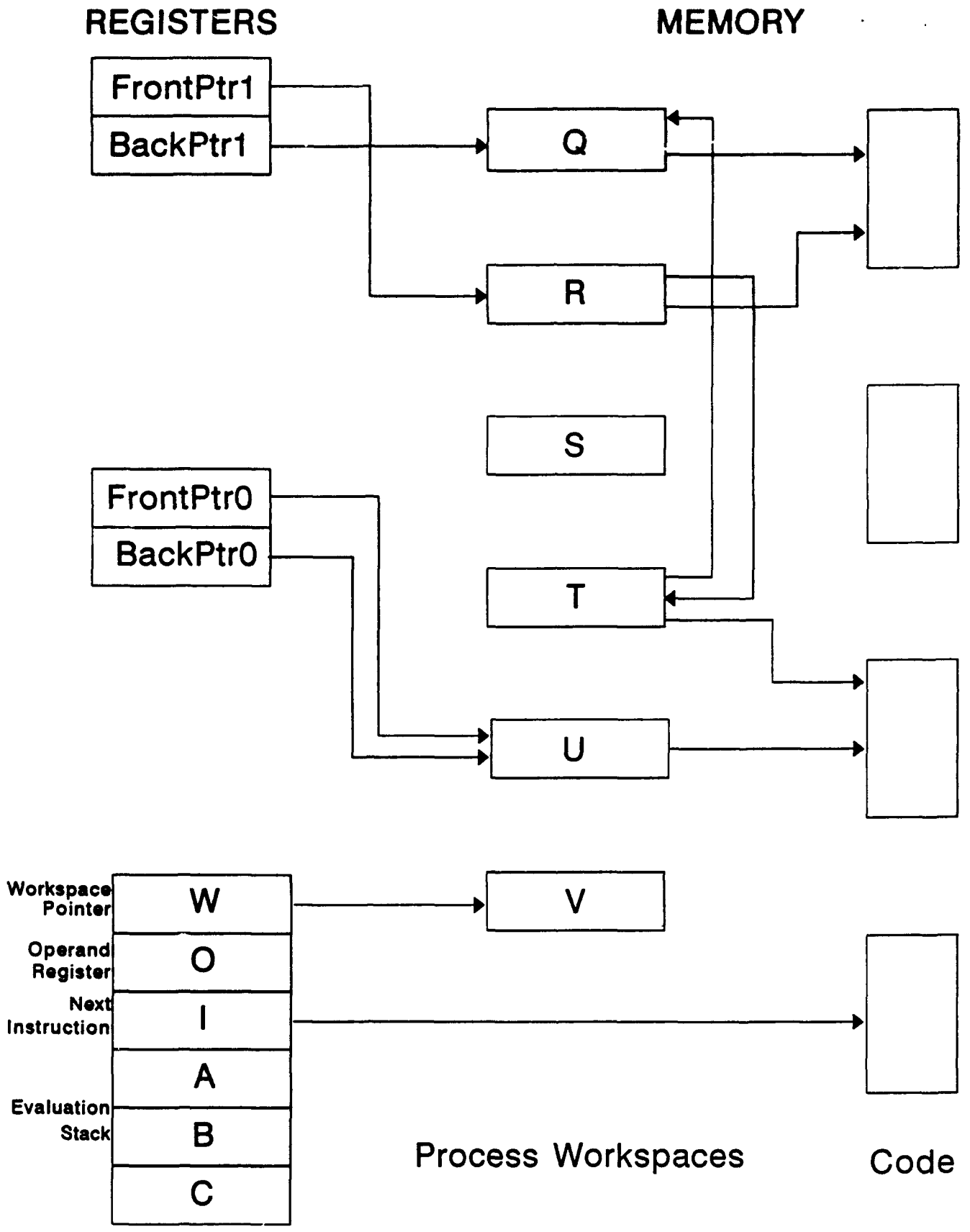

Figure 2: Transputer Registers

Transputer Registers

The register model for the transputer is shown in Figure 2 on the previous page. All registers are either 16 bits or 32 bits, depending on the word length of the transputer. Registers A, B, and C form an evaluation stack. Instead of using general purpose registers, transputer instructions are designed around the use of this stack. A stack depth of three is used to provide a good compromise between the ability to to evaluate most expressions on the stack and having as little as possible to save when there is a context switch. Register W is the workspace pointer -- a pointer into the local variables associated with the currently executing process. Many instructions refer to data by its offset from the workspace pointer. The instruction pointer register (I) points to the next instruction to be executed. This is equivalent to the program counter (PC) in conventional microprocessors. The construction of operands is the function of the operand or O register.

The Floating-Point Unit

The T8 series of transputers has a 32/64-bit floating point unit that conforms to the IEEE 754-1985 specification. The floating point unit (FPU) has an evaluation stack similar to that of the integer processor, with three registers -- FA, FB, and FC. Each of these registers can contain either a 32-bit or a 64-bit number and has a flag to show which of these it does conatin. The FPU design is a compromise between maximizing overall processor performance and minimizing chip area. Because of this, the FPU has no flash multiplier or barrel shifter. However, the performance is good, with single and double precision multiplication times of 550 and 1000 nanoseconds respectively, for a 20 MHz device. The FPU operates concurrently with the integer processor, and thus computation can be speeded up by overlapping integer and floating-point processing. Figure 3 shows the floating point unit.

The Instruction Set (Occam)

The transputer does not readily fall into either the CISC (complex instruction set) or the RISC (reduced instruction set) categories. It has a simple instruction set, called Occam, and tends to be viewed as a RISC processor. However, it is much more than a RISC processor because of the functionality built into the chip to support high level concepts such as processes, timers, and inter-process communication. Programmers used to programming other microprocessors may find programming the transputer to be a strange experience. There is only a small number of registers that are organized as a stack, and all instructions are stack, rather than register oriented. There is little concept of condition codes, only limited instructions for accessing memory, and more sophisticated memory-addressing modes.

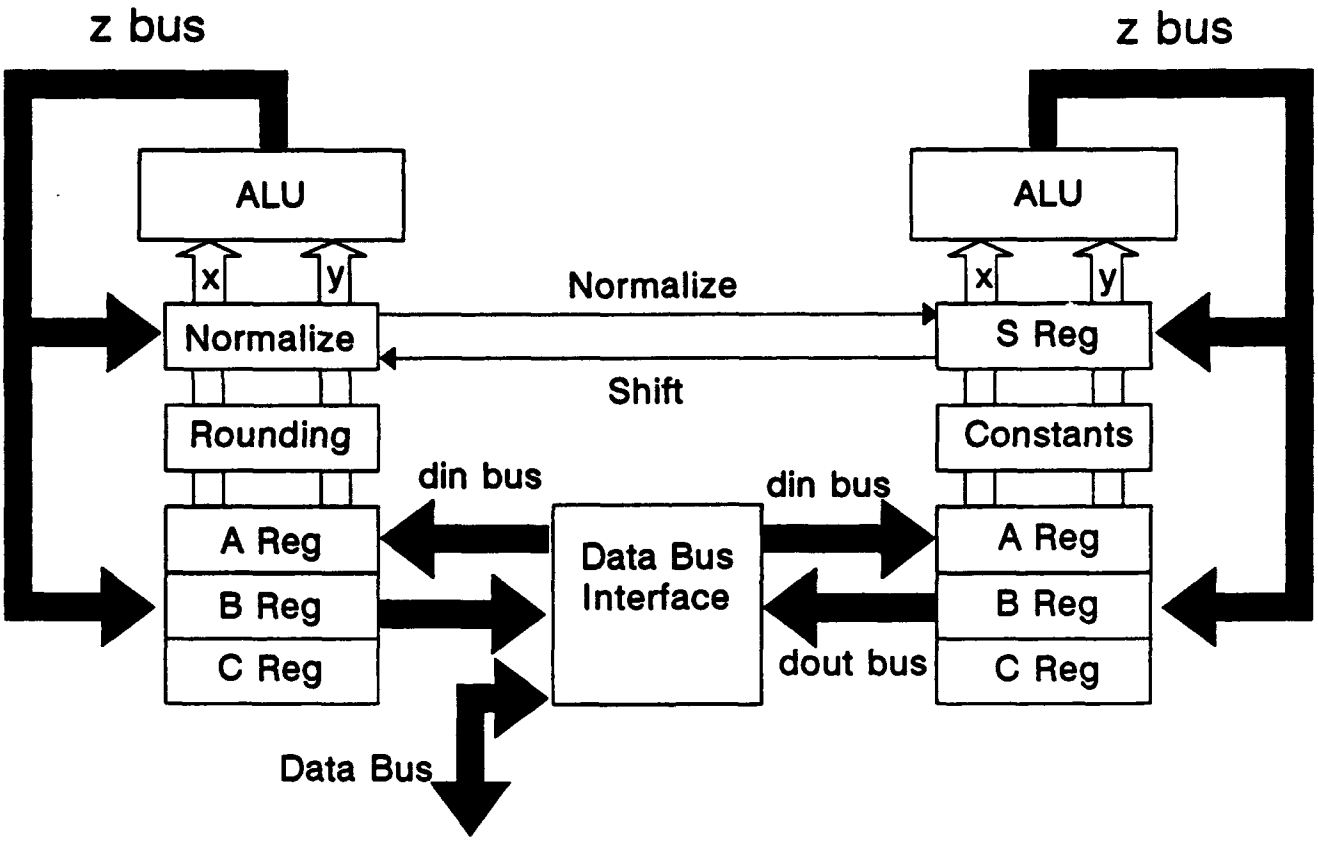

Figure 3: Floating Point Unit

processor cycles. There is prefetch queue which holds eight instructions and is purged whenever a jump occurs.

Occam enables a system to be described as a collection of concurrent processes that communicate with each other and with periphral devices through channels. All Occam programs are built form these three primitives:

v := e	assign expression **e** to variable **v**
c ! e	output expression **e** to channel **c**
c ? v	input from channel **c** to variable **v**

These primitive processes are combined to form constructs such as:

SEQential	components executed one after the other
PARallel	components executed together
ALTernative	component first ready is executed

A construct is itself a process, and ma be used as a component of another construct. Variables and assignments combined in sequential constructs can be used to express conventional sequential programs. **IF** and **WHILE** constructs are also available. Concurrent processes can be expressed with channels, inputs, and outputs, which are combined in parallel and alternative constructs.

Each Occam channel provides a communication path between two concurrent processes. Communication is synchronized and takes place only when both the sending and receiving process are ready. The data is then copied from the sending process to the receiving process and both processes continue. An alternative process may be ready for input from any one of a number of channels. When this happens, input is taken from the first channel that is used for output by another process.

0	Jump	8	Add Constant
1	Load Local Pointer	9	Call
2	Prefix	10	Conditional Jump
3	Load Non-Local	11	Adjust Workspace
4	Load Constant	12	Equals Constant
5	Load Non-Local Pointer	13	Store Local
6	Negative Prefix	14	Store Non-Local
7	Load Local	15	Operate

Table 2 - Primary Transputer Instructions

Transputer RAM

The transputer has on-chip RAM that can be used in exactly the same manner as external memory, except that accesses to internal memory are at least two to three times faster. This is unlike a cache in that the user must make explicit use of the on-chip memory and must decide what it is to contain. Protection of the address space of one process from the actions of another can be achieved by putting the two processes on separate transputers, rather than by a memory management algorithm.

All of the present transputers have either two kilobytes or four kilobytes of on-chip fast static RAM that occupies the very bottom of the memory space. Some of this is reserved for processor microcode functions. Two kilobytes or four kilobytes is not adequate for most funtions, so there is an external memory interface. Reads and writes to internal memory are fast, requiring only a single clock cycle. External memory accesses take at least two processor cycles, but usually take four or five in dynamic memory systems. Memory addresses are signed, so the memory of a transputer starts at the lowest possible negative number -- **MinInt**. **MinInt** is 0x8000 on a 16 bit transputer or 0x80000000 on a 32 bit transputer. Memory runs from **MinInt** to **MaxInt**, which is 0x7FFF on 16 bit processors or 0x7FFFFFFF on the 32 bit processors.

Table 3 shows the reserved memory addresses and their uses. The addresses are shown as word offsets from **MinInt**, with the actual byte address dependent on the word length of the transputer. The lowest eight words contain the the channel control words for the four external links, and the event control word is in the ninth location. The next two positions contian the the front pointers for the high and low priority timer process queues. The next seven words store the processor state when when a low priority process is interrupted by a high priority process. This is the only time the state must be saved.

Interrupts

The only source of external interrupts is the **EventReq** input. The programming interface is implemented so that it appears as another control word located in low memory. If a processes needs to wait on **EventReq**, it will be desceduled until the line goes high, and will then be rescheduled. When this happens, the event output handshake line **EventAck** is driven high by the processor. In order to provide rapid servive to interrupts, the process waiting for the event input must be high priority, and be the only high priority process running. Interrupt service time is made faster by making long intructions interruptable.

Word Address	Name	Use
MinInt+28	MemStart	T805, T801, T800, T425, T225
...		
...		reserved for extended
...		instructions
...		(not T212, M212, T414)
...		
MinInt+18	MemStart	T414, T212, M212
MinInt+17	EregIntSaveLoc	
MinInt+16	STATUSIntSavLoc	
MinInt+15	CregIntSaveLoc	Register Save Area
MinInt+14	BregIntSaveLoc	
MinInt+13	AregIntSaveLoc	
MinInt+12	IptrIntSaveLoc	
MinInt+11	WdescIntSaveLoc	
MinInt+10	TprtLoc1	low-priority timer
MinInt+9	TptrLoc0	high-priority timer
MinInt+8	Event	Event Timer
MinInt+7	Link3Input	
MinInt+6	Link2Input	
MinInt+5	Link1Input	
MinInt+4	Link0Input	link control
MinInt+3	Link3Output	words
MinInt+2	Link2Output	
MinInt+1	Link1Output	
MinInt	Link0Output	

Table 3 - Reserved memory locations.

Operation

The transputer contains a unique microcoded process scheduler which maintains two process queues, one at a high priority and one at a low priority. Processes in the high priority queue are allowed to execute until termination or until they require input or output. Low priority processes are automatically time sliced at intervals of about one millisecond and can be interrupted by high priority processes. The queues are implemented as linked lists through the workspace of the active processes. The front and back of each queues are pointed to by two registers, one for each priority. There is a timer and a timer process queue associated with each priority level. The timer registers can be read directly or the scheduler can be instructed to queue a process for execution when a timer reaches a certain value. The

scheduler has the ability to select a group of processes for execution based on the occurrence of some event such as completion of data input, expiration of a timer period, or external interrupt.

The transputer has a system services interface that includes signals necessary to reset and boot itself, set the speed of its processor and its links, to signal errors, and to respond to external events. Based upon the value of the **Analyse** signal at reset, the transputer can be reset into one of two modes. If **Analyse** is low at reset, the transputer will boot from memory if the signal **BootFromRom** is high. If **BootFromRom** is low, the transputer will accept from its links either a bootstrap program or commands to read or write memory locations. **Analyse** can be used as a debugging tool. If **Analyse** is taken high during operation, the transputer will soon halt, and the links will become inactive. Upon reset, the registers will contain information about the state of the machine when it was halted by taking **Analyse** high.

Communications

The major extra function built into all members of the transputer family is on chip communications. This makes the transputer unique because it comes equipped with the ability to communicate with other transputers. This makes it easy to to construct arrays of processors that work together as a MIMD computer. The transputer has been designed for message passing in parallel computers, so has strong inter-process communication capabilities. This interface has been designed so that there is almost no difference to the programmer -- whether the communication is between two processes executing on the same or different transputers. The basic unit of information transfer between transputers is the link. A link is comprised of two channels -- an input channel and an output channel. Information is transferred between two processes on the same transputer by copying data between their memory spaces. The transfer is synchronized using a channel control word, which is a single word anywhere in memory. Communication takes place only when both the sending and receiving porcesses are ready. A process waiting to communicate will use no processor time. Two transputer processes can synchronize by simply passing a message between them. Communication between processes on different transputers uses the same mechanism, but is synchronized by special channel control words that reside in reserved locations in low memory. The same instructions are used to set up the transfer, and the link engines in each transputer take care of the DMA transfer between link and memory, so no processor intervention is necessary.

All of the presently available transputers have either two or four external links. These links are full duplex and can exchange data with other transputers at 5, 10, or 20 million bits per second. The bidirectional data rate is dependent on the types of processors exchanging data and the link speed. The speed can be up to 2.4 Mbytes per second on T800 transputers with 20 Mbps links. The link data is transferred as a serial byte stream, with each byte acknowledged by the receiving transputer. No hardware attempt is made to detect errors, so

it is important that the link be made with a reliable physical meduim. The quiescent sate of a link output is low. Each data byte is transmitted as a high start bit, followed by a one bit, followed by eight data bits, followed by a low stop bit. The least significant bit of dtat is transmitted first. After transmitting a data byte, the sender waits for an acknowledge, which consists of a high start bit followed by a zero bit. The acknowledge signifies that both that a process was able to receive the acknowledged data byte and that the receiver is ready to receive another data byte. The sending link reschedules the sending process only after the acknowledge for the final byte of the message has been received.

Problems

Although the transputer has many good qualities, it also has several problems. There is no support for memory management or virtual memory. This lack is a result of the transputer design goals. It was designed for implementation of multiprocessor systems, with each processor having its own local memory and communicating by message passing along fast links. Also, the on-chip fast RAM is not a cache, which is a speed impediment. The design will be hard to scale past the 25 MHz to 30 MHz range. While this was fast when the transputer was introduced in 1984, it is by no means fast today. The transputer has only four links, so the most complex topology for interconnection is a 2-D mesh, no 3-D patterns such as the hypercube are possible.

Because of these problems, and addiditional problems, such as poor support and upgrades that are behind schedule, the future of the transputer looks bleak. The T9000 transputer was due to be released in May of 1991, but still has not been released. Faster processors, such as the DEC Alpha may make the T9000 obsolete before it is even in silicon. It now appears that the next generation of transputer will be less accepted outside of Europe than its predecessors. Even with all of its problems, the transputer has helped change the way computers are designed. So it has at least been successful in helping speed up the advance of microprocessor technology.

Chemical Defense Applications

The transputer has several potential applications in the field of chemical defense. In fact, the Chemical-Biological Mass Spectrometer (CBMS), currently under development at Edgewood Research, Development, and Engineering Center (ERDEC), uses four transputers. These transputers will simultaneously be performing different functions necessary for the operation of the instrument.

The Biological Integrated Defense System (BIDS), being developed by ERDEC, shows great potential for the use of transputers. The BIDS provides a sophisticated point detection ability by integrating biological detection instruments, sampler/concentrators, and personnel within a protective enclosure mounted on a high mobility vehicle. This type of problem lends

itself to parallel processing. Not only must the output of each instrument be simultaneously processed, but each instrument must be controlled concurrently. Another application with great potential includes instruments with multiple detection schemes. An example of this is the Coulter Flow Cytometer. There is a great amount of flexibility in the four sensor inputs of the flow cytometer. Transputers could be used to partition incoming data in whatever manner will result in the best data analysis. For example, input data from each sensor could be processed in parallel on separate transputers. When a result from a calculation on one transputer is needed by another transputer, they can simply communicate via their links and continue processing.

Transputers should only be considered for these systems if the inherent parallelism of the transputer architecture is to be exploited. In each of these systems, it is important for separate processing to be occurring and some communication to take place between the different processors.

Conclusion

The Inmos transputer family is a range of system components which can be used to construct high performance concurrent systems. As all members of the family incorporate Inmos communications links, a system can be constructed from different members of the family. All transputers provide hardware support for concurrency and offer exceptional performance on process scheduling, inter-process communication, and inter-transputer communication.

The design of transputers is very economical in the use of silicon. The central processor used in the transputer offers a performance comparable to other VLSI processors that are several times larger. The small size of the processor allows a memory and a communications system to be integrated on the same VLSI device. This level of integration allows very fast access to memory and very fast inter-transputer communication. Similarly, the transputer floating point unit is integrated into the same device as the central processor, eliminating the delays inherent in communicating data between the devices.

LITERATURE CITED

1. **The Transputer Handbook**, Ian graham & Tim King, Prentice Hall International, 1990.

2. **The Transputer Databook**, INMOS, 1989.

3. **Inside the Transputer**; D.A. P. Mitchell, J.A. Thompson, G.A. Manson, G.R. Brooks; Blackwell Scientific Publications; 1990.

4. Lecture Notes, Loyola College, EG 739 - Computer Organization, March 20, 1993.